PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-240741

(43)Date of publication of application : 11.09.1998

(51)Int.Cl. G06F 17/30

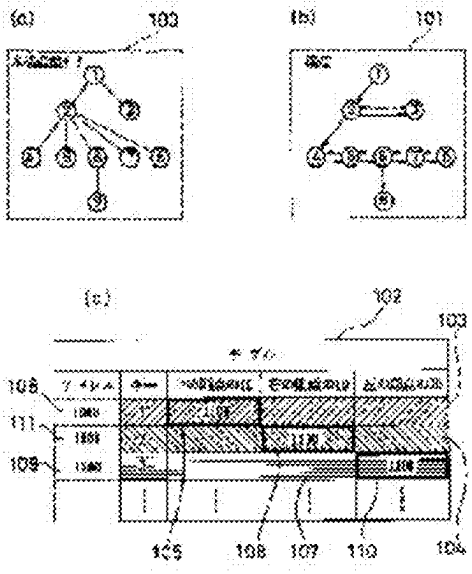
(21)Application number : 09-046368

(22)Date of filing : 28.02.1997

(71)Applicant : NIPPON TELEGR & TELEPH CORP <NTT>

(72)Inventor : KONISHI FUMIKAZU
KOBAYASHI NOBUYUKI
NATSUME YOSHIHISA
NISHIOKA SHUICHI

(54) MANAGING METHOD FOR TREE STRUCTURE TYPE DATA



(57)Abstract:

PROBLEM TO BE SOLVED: To directly obtain the position of a physical record by using an ID when a child of an arbitrary node are retrieved by representing the parent-child relation of a tree structure by using IDs representing the storage positions where respective nodes are stored.

SOLUTION: To store tree structure type data 100 in a relational data base, the number of branches between the parent and children is fixed to a finite number and the relation between the parent and children is deformed into a structure 101. Then data corresponding to nodes 1, 2, and 3 of the tree structure are stored as records 103, 104, and 107 of the relational data base in a table 102, and IDs 1100, 1200, and 1300 indicating the storage positions of the stored records are

provided and stored in columns 105, 108, and 110 of the records. Consequently, one node requires up to three branches (up arrow, right array, and left array) like a node 6 and the number of the branches is fixed. Consequently, fast retrieval of the tree structure data becomes possible.

LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

CLAIMS

[Claim(s)]

[Claim 1] The data which hit at the joint of the tree structure in the processing which stores tree structure mold data in a relational database are stored as a record of a relational database. The management method of the tree structure mold data characterized by limiting the branch between two or more children who prepare ID showing the storing location of this stored record, store this ID in the column of a record, and receive one parents in the tree structure, and its parents to the branch between the children of the finite individual of the arbitration in said two or more children.

[Claim 2] The management method of the tree structure mold data according to claim 1 characterized by searching the record which exists in the storing location shown by said ID in case the child of a joint to said joint of the arbitration in the tree structure is searched from the relational database which stored tree structure mold data.

[Translation done.]

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Field of the Invention] This invention relates to the management method of the tree structure mold data with which this stored tree structure mold data is searched while storing tree structure mold data.

[0002]

[Description of the Prior Art] Tree structure mold data are data which have structure as shown in drawing 3 . In this drawing, each thing expressed with a circle is called a joint (top-most vertices). One in a joint is called a root (root) [else]. A parentage

exists between joints and it is connected with the line called a branch (side) between the joints of parents and a child. The joints of a branch top are parents and a lower joint is a child. Only one has parents in each joint other than a root.

[0003] On the other hand, there is no limit in the number of the children of a joint, and n children's branch may exist to one parents. Each joint may have how old child's joint, and there may be a joint without a child. It is tree structure mold data which connected all the joints with the branch according to such a regulation.

[0004] When treating such tree structure mold data by the relational database, there is an approach shown below.

[0005] As the 1st approach, as shown in drawing 4 , the whole tree structure is expressed as one table. That is, the vertical relation of the tree structure is expressed by storing in a column in order the key of all the joints that exist in the route to which one joint on the tree structure is expressed, the tree structure prepares a column for a table by the number of stages, and one record of a table reaches a certain joint originally.

[0006] As the 2nd approach, as shown in drawing 5 , the key for identifying uniquely within the tree structure is set to each joint of the tree structure, each record of a table expresses each joint, it is storing in the column of a table the key of the joint of the parents who are the key of each joint, i.e., their own key, and the key of the joint of the direct high order, respectively, and the vertical relation to the tree structure is expressed.

[0007]

[Problem(s) to be Solved by the Invention] There were the following troubles in the management method of the conventional tree structure mold data mentioned above.

[0008] (1) If the 1st approach is used, there will be little data base manipulation required for retrieval of the tree structure, and it will end, but since it is necessary to prepare all number-of-stages part columns for a table beforehand, the storing effectiveness of data is very bad, including mostly the column which is not used. Moreover, since a wooden number of stages may increase with insertion of data and the number of columns of a table changes in that case, the need of remaking a database comes out.

[0009] (2) If the 2nd approach is used, in order to search m steps of tree structures, the data base manipulation (reference actuation) to m times of tables is required. When the number of stages of the tree structure is deep, remarkable retrieval time is required.

[0010] This invention was made in view of the above, and the place made into the

purpose is to offer the management method of the tree structure mold data which can store tree structure mold data efficiently while the high-speed retrieval to tree structure mold data is possible.

[0011]

[Means for Solving the Problem] In order to attain the above-mentioned purpose, this invention according to claim 1 The data which hit at the joint of the tree structure in the processing which stores tree structure mold data in a relational database are stored as a record of a relational database. ID showing the storing location of this stored record is prepared, and this ID is stored in the column of a record, and let it be a summary to limit the branch between two or more children who receive one parents in the tree structure, and its parents to the branch between the children of the finite individual of the arbitration in said two or more children.

[0012] If it is in this invention according to claim 1, the data which hit at each joint in tree structure mold data are stored in the record of a relational database, and ID showing the storing location of this stored record is acquired, it stores in the column of a record, and the parentage in the tree structure is expressed by ID. Moreover, in order to express the parentage in the tree structure by the relational database A branch with two or more children who receive the parents and parents at the time of making the joint of arbitration into parents is limited to a finite individual. The column for storing ID for the finite number is prepared for a record, the branch of a finite individual is mutually defined among two or more children which receive the parents at the time of making the joint of arbitration into parents, the column for storing ID of this ramification is formed in a record, and the tree structure is stored in a relational database.

[0013] Moreover, in invention according to claim 1, in case this invention according to claim 2 searches the child of a joint to said joint of the arbitration in the tree structure from the relational database which stored tree structure mold data, it makes it a summary to search the record which exists in the storing location shown by said ID.

[0014] If it is in this invention according to claim 2, when performing retrieval to the child's joint of arbitration from a joint (record) to the tree structure mold data stored in the relational database, ID is used for the location to which said ID points, and a direct relevance record is searched.

[0015]

[Embodiment of the Invention] Hereafter, the gestalt of operation of this invention is explained using a drawing.

[0016] With reference to drawing 1 , the management method of the tree structure

mold data concerning 1 operation gestalt of this invention is explained.
[0017] First, how to store in a relational database the tree structure mold data 100 shown in drawing 1 (a) is explained.
[0018] When five columns are needed in order to store the branch to the child of a joint 2 in a table when there are five branches like a joint 2, and there are two branches like a joint 1, in order to store the branch to the child of a joint 1 in a table, two columns will be needed, and the required number of columns will change according to the number of the branches to the child whom a joint has.
[0019] Then, the branch between parents and a child is fixed to a finite individual, and it is made to deform with this operation gestalt like the structure 101 which shows the relation between parents and a child to the tree structure mold data 100 in drawing 1 (b). By this, the number of the branches which one joint needs becomes a maximum of three (an arrow down, a right arrow, left arrow) like a joint 6, and it becomes possible to fix the number of branches.
[0020] Thus, how to store in the table 102 showing the structure 101 which can be stored now in the column of a finite individual in drawing 1 (c) is explained.
[0021] The column which stores the column which stores the column which stores in a table 102 the column which stores a key, and ID which points out the storing location of the record with which the lower joint is stored, and ID which points out the storing location of the record with which the right joint is stored, and ID which points out the storing location of the record with which the left joint is stored is prepared. However, although ID which shall arrange the table on memory and points out the storing location of a record considers as a memory address, this example will be available for it anything, if it seems that ID which points out a storing location can express the location stored directly.
[0022] The step of the processing which stores the data of structure 101 in a table 102 is as follows.
[0023] (Step S1) The key of a joint 1 is stored in the record 103 of a table 102.
[0024] (Step S2) The key of a joint 2 is stored in the record 104 of a table 102.
[0025] (Step S3) The memory address 111 of a record 104 is stored in ID storing column 105 of the joint under a record 103.
[0026] (Step S4) The key of a joint 3 is stored in the record 107 of a table 102.
[0027] (Step S5) The memory address 109 of a record 107 is stored in ID storing column 108 of the joint on the right of a record 104.
[0028] (Step S6) The memory address 111 of a record 104 is stored in ID storing column 110 of the joint on the left of a record 107.

[0029] The above-mentioned step is performed about all the joints of structure 101. However, like a joint 8, when the right, the left, or a lower joint does not exist, the value which that a joint does not exist any more can identify is stored.
[0030] Next, the search method for a table 102 is explained.
[0031] The step of the processing which searches the joint 3 (a key is set to 3) which is the child of a joint 1 (a key is set to 1) to a table 102 is as follows.
[0032] (Step S1) It jumps on a record 104 using the memory address 105 stored in ID storing column of the lower joint of the record 103 with which the joint 1 is stored.
[0033] (Step S2) The key 2 of a record 104 is compared with the key 3 of the given joint 3.
[0034] (Step S3) Since a key is not in agreement, it jumps on a record 107 using the memory address 109 stored in ID storing column 108 of the right joint of a record 104.
[0035] (Step S4) The key 3 of a record 107 is compared with the key 3 of the given joint 3.
[0036] (Step S5) Since the key was in agreement, retrieval processing is ended.
[0037] It continues until the target joint finds the above-mentioned step. When the value which that the joint does not exist any more [above] can identify is stored in ID storing column of a right joint, it means that the joint which is in agreement with the key does not exist in a database.
[0038] Next, other operation gestalten of this invention are explained with reference to drawing 2 .
[0039] How to store in a relational database the tree structure mold data 200 shown in drawing 2 (a) is explained.
[0040] Since the branch between parents and a child is fixed to a finite individual, it is made to deform in this operation gestalt like the structure 201 which shows the relation between parents and a child to the tree structure mold data 200 in drawing 2 (b). By this, the number of the branches which one joint needs becomes a maximum of five (a lower left arrow head, a bottom middle arrow head, a lower right arrow head, a right arrow, left arrow) like a joint 3, and it becomes possible to fix the number of branches.
[0041] Next, the method of storing in a table the structure 201 which became as [store / by the above / it / in the column of a finite individual] is the same as that of the operation gestalt of drawing 1 . However, rearrangement is beforehand performed for the key from the joint 5 in structure 201 to a joint 9 in ascending order or descending order.
[0042] Next, the search method for a table is the same as that of the operation gestalt

of drawing 1 . However, in case the child is searched from the joint of arbitration, after comparing by the key value of the joint which a bottom middle arrow head points out, it determines whether to jump on the record of three downward ID.

[0043] High-speed retrieval is enabled in jumping directly on the record which ID points out, without performing retrieval via an index, when performing retrieval to a tree by storing each joint in the tree structure in a record, and storing the branch during each joint (relation) in the column of a record in the management method of the tree structure mold data of this invention as ID which points out the storing location of a record, as mentioned above.

[0044] Moreover, by limiting the branch from parents to a child for the parentage of 1:n in the tree structure to a finite individual, and connecting children with a branch mutually, since the column of a finite (immobilization) individual expresses the tree structure, the column which is not used is lost and efficient storing can be performed.

[0045]

[Effect of the Invention] Since it is expressing using ID showing the storing location where the parentage in the tree structure is stored in each knot according to this invention as explained above, when searching the child of the joint of arbitration, it becomes possible to acquire the location of a physical record by using said ID, and enables this to ask for said record directly. Therefore, it is not necessary to use data base manipulation (reference actuation) for retrieval of tree structure mold data.

[0046] Moreover, since the branch between two or more children who receive said parents and said parents at the time of making the joint of arbitration into parents can be limited to a finite individual and the branch between two or more children which receive said parents can also be limited to a finite individual, the column of a column required for a table which do not use by being made to immobilization regardless of the number of stages of the tree structure is lost, and storing effectiveness becomes very good.

[0047] When the number of stages of the tree structure increases, it becomes unnecessary moreover, to newly remake a database, since a column required for a table is the number of immobilization (the above-mentioned finite individual).

[Translation done.]

* NOTICES *

TECHNICAL FIELD

[Field of the Invention] This invention relates to the management method of the tree structure mold data with which this stored tree structure mold data is searched while storing tree structure mold data.

[Translation done.]

* NOTICES *
JPO and NCIPI are not responsible for any
damages caused by the use of this translation.
1.This document has been translated by computer. So the translation may not reflect the original precisely.
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

PRIOR ART

[Description of the Prior Art] Tree structure mold data are data which have structure as shown in drawing 3 . In this drawing, each thing expressed with a circle is called a joint (top-most vertices). One in a joint is called a root (root) [else]. A parentage exists between joints and it is connected with the line called a branch (side) between the joints of parents and a child. The joints of a branch top are parents and a lower joint is a child. Only one has parents in each joint other than a root.
[0003] On the other hand, there is no limit in the number of the children of a joint, and n children's branch may exist to one parents. Each joint may have how old child's joint,

and there may be a joint without a child. It is tree structure mold data which connected all the joints with the branch according to such a regulation.

[0004] When treating such tree structure mold data by the relational database, there is an approach shown below.

[0005] As the 1st approach, as shown in drawing 4 , the whole tree structure is expressed as one table. That is, the vertical relation of the tree structure is expressed by storing in a column in order the key of all the joints that exist in the route to which one joint on the tree structure is expressed, the tree structure prepares a column for a table by the number of stages, and one record of a table reaches a certain joint originally.

[0006] As the 2nd approach, as shown in drawing 5 , the key for identifying uniquely within the tree structure is set to each joint of the tree structure, each record of a table expresses each joint, it is storing in the column of a table the key of the joint of the parents who are the key of each joint, i.e., their own key, and the key of the joint of the direct high order, respectively, and the vertical relation to the tree structure is expressed.

[Translation done.]

EFFECT OF THE INVENTION

[Effect of the Invention] Since it is expressing using ID showing the storing location where the parentage in the tree structure is stored in each knot according to this invention as explained above, when searching the child of the joint of arbitration, it becomes possible to acquire the location of a physical record by using said ID, and

enables this to ask for said record directly. Therefore, it is not necessary to use data base manipulation (reference actuation) for retrieval of tree structure mold data.
[0046] Moreover, since the branch between two or more children who receive said parents and said parents at the time of making the joint of arbitration into parents can be limited to a finite individual and the branch between two or more children which receive said parents can also be limited to a finite individual, the column of a column required for a table which do not use by being made to immobilization regardless of the number of stages of the tree structure is lost, and storing effectiveness becomes very good.
[0047] When the number of stages of the tree structure increases, it becomes unnecessary moreover, to newly remake a database, since a column required for a table is the number of immobilization (the above-mentioned finite individual).

[Translation done.]

TECHNICAL PROBLEM

[Problem(s) to be Solved by the Invention] There were the following troubles in the management method of the conventional tree structure mold data mentioned above.
[0008] (1) If the 1st approach is used, there will be little data base manipulation required for retrieval of the tree structure, and it will end, but since it is necessary to prepare all number-of-stages part columns for a table beforehand, the storing effectiveness of data is very bad, including mostly the column which is not used. Moreover, since a wooden number of stages may increase with insertion of data and the number of columns of a table changes in that case, the need of remaking a

database comes out.
[0009] (2) If the 2nd approach is used, in order to search m steps of tree structures, the data base manipulation (reference actuation) to m times of tables is required. When the number of stages of the tree structure is deep, remarkable retrieval time is required.
[0010] This invention was made in view of the above, and the place made into the purpose is to offer the management method of the tree structure mold data which can store tree structure mold data efficiently while the high-speed retrieval to tree structure mold data is possible.

[Translation done.]

MEANS

[Means for Solving the Problem] In order to attain the above-mentioned purpose, this invention according to claim 1 The data which hit at the joint of the tree structure in the processing which stores tree structure mold data in a relational database are stored as a record of a relational database. ID showing the storing location of this stored record is prepared, and this ID is stored in the column of a record, and let it be a summary to limit the branch between two or more children who receive one parents in the tree structure, and its parents to the branch between the children of the finite individual of the arbitration in said two or more children.
[0012] If it is in this invention according to claim 1, the data which hit at each joint in tree structure mold data are stored in the record of a relational database, and ID showing the storing location of this stored record is acquired, it stores in the column

of a record, and the parentage in the tree structure is expressed by ID. Moreover, in order to express the parentage in the tree structure by the relational database A branch with two or more children who receive the parents and parents at the time of making the joint of arbitration into parents is limited to a finite individual. The column for storing ID for the finite number is prepared for a record, the branch of a finite individual is mutually defined among two or more children which receive the parents at the time of making the joint of arbitration into parents, the column for storing ID of this ramification is formed in a record, and the tree structure is stored in a relational database.

[0013] Moreover, in invention according to claim 1, in case this invention according to claim 2 searches the child of a joint to said joint of the arbitration in the tree structure from the relational database which stored tree structure mold data, it makes it a summary to search the record which exists in the storing location shown by said ID.

[0014] If it is in this invention according to claim 2, when performing retrieval to the child's joint of arbitration from a joint (record) to the tree structure mold data stored in the relational database, ID is used for the location to which said ID points, and a direct relevance record is searched.

[0015]

[Embodiment of the Invention] Hereafter, the gestalt of operation of this invention is explained using a drawing.

[0016] With reference to drawing 1 , the management method of the tree structure mold data concerning 1 operation gestalt of this invention is explained.

[0017] First, how to store in a relational database the tree structure mold data 100 shown in drawing 1 (a) is explained.

[0018] When five columns are needed in order to store the branch to the child of a joint 2 in a table when there are five branches like a joint 2, and there are two branches like a joint 1, in order to store the branch to the child of a joint 1 in a table, two columns will be needed, and the required number of columns will change according to the number of the branches to the child whom a joint has.

[0019] Then, the branch between parents and a child is fixed to a finite individual, and it is made to deform with this operation gestalt like the structure 101 which shows the relation between parents and a child to the tree structure mold data 100 in drawing 1 (b). By this, the number of the branches which one joint needs becomes a maximum of three (an arrow down, a right arrow, left arrow) like a joint 6, and it becomes possible to fix the number of branches.

[0020] Thus, how to store in the table 102 showing the structure 101 which can be

stored now in the column of a finite individual in drawing 1 (c) is explained.
[0021] The column which stores the column which stores the column which stores in a table 102 the column which stores a key, and ID which points out the storing location of the record with which the lower joint is stored, and ID which points out the storing location of the record with which the right joint is stored, and ID which points out the storing location of the record with which the left joint is stored is prepared. However, although ID which shall arrange the table on memory and points out the storing location of a record considers as a memory address, this example will be available for it anything, if it seems that ID which points out a storing location can express the location stored directly.
[0022] The step of the processing which stores the data of structure 101 in a table 102 is as follows.
[0023] (Step S1) The key of a joint 1 is stored in the record 103 of a table 102.
[0024] (Step S2) The key of a joint 2 is stored in the record 104 of a table 102.
[0025] (Step S3) The memory address 111 of a record 104 is stored in ID storing column 105 of the joint under a record 103.
[0026] (Step S4) The key of a joint 3 is stored in the record 107 of a table 102.
[0027] (Step S5) The memory address 109 of a record 107 is stored in ID storing column 108 of the joint on the right of a record 104.
[0028] (Step S6) The memory address 111 of a record 104 is stored in ID storing column 110 of the joint on the left of a record 107.
[0029] The above-mentioned step is performed about all the joints of structure 101. However, like a joint 8, when the right, the left, or a lower joint does not exist, the value which that a joint does not exist any more can identify is stored.
[0030] Next, the search method for a table 102 is explained.
[0031] The step of the processing which searches the joint 3 (a key is set to 3) which is the child of a joint 1 (a key is set to 1) to a table 102 is as follows.
[0032] (Step S1) It jumps on a record 104 using the memory address 105 stored in ID storing column of the lower joint of the record 103 with which the joint 1 is stored.
[0033] (Step S2) The key 2 of a record 104 is compared with the key 3 of the given joint 3.
[0034] (Step S3) Since a key is not in agreement, it jumps on a record 107 using the memory address 109 stored in ID storing column 108 of the right joint of a record 104.
[0035] (Step S4) The key 3 of a record 107 is compared with the key 3 of the given joint 3.
[0036] (Step S5) Since the key was in agreement, retrieval processing is ended.

[0037] It continues until the target joint finds the above-mentioned step. When the value which that the joint does not exist any more [above] can identify is stored in ID storing column of a right joint, it means that the joint which is in agreement with the key does not exist in a database.

[0038] Next, other operation gestalten of this invention are explained with reference to drawing 2 .

[0039] How to store in a relational database the tree structure mold data 200 shown in drawing 2 (a) is explained.

[0040] Since the branch between parents and a child is fixed to a finite individual, it is made to deform in this operation gestalt like the structure 201 which shows the relation between parents and a child to the tree structure mold data 200 in drawing 2 (b). By this, the number of the branches which one joint needs becomes a maximum of five (a lower left arrow head, a bottom middle arrow head, a lower right arrow head, a right arrow, left arrow) like a joint 3, and it becomes possible to fix the number of branches.

[0041] Next, the method of storing in a table the structure 201 which became as [store / by the above / it / in the column of a finite individual] is the same as that of the operation gestalt of drawing 1 . However, rearrangement is beforehand performed for the key from the joint 5 in structure 201 to a joint 9 in ascending order or descending order.

[0042] Next, the search method for a table is the same as that of the operation gestalt of drawing 1 . However, in case the child is searched from the joint of arbitration, after comparing by the key value of the joint which a bottom middle arrow head points out, it determines whether to jump on the record of three downward ID.

[0043] High-speed retrieval is enabled in jumping directly on the record which ID points out, without performing retrieval via an index, when performing retrieval to a tree by storing each joint in the tree structure in a record, and storing the branch during each joint (relation) in the column of a record in the management method of the tree structure mold data of this invention as ID which points out the storing location of a record, as mentioned above.

[0044] Moreover, by limiting the branch from parents to a child for the parentage of 1:n in the tree structure to a finite individual, and connecting children with a branch mutually, since the column of a finite (immobilization) individual expresses the tree structure, the column which is not used is lost and efficient storing can be performed.

[Translation done.]

* NOTICES *

JPO and NCIPI are not responsible for any
damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.

2.**** shows the word which can not be translated.

3.In the drawings, any words are not translated.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1] It is drawing for explaining the management method of the tree structure mold data concerning 1 operation gestalt of this invention.

[Drawing 2] It is drawing for explaining other operation gestalten of this invention.

[Drawing 3] It is drawing for explaining tree structure mold data.

[Drawing 4] It is drawing for explaining the management method of the conventional tree structure mold data.

[Drawing 5] It is drawing for explaining the management method of the conventional tree structure mold data.

[Description of Notations]

100,200 Tree structure mold data

101,201 Tree structure mold data after conversion

102 Table

103 Record Which Stores Joint 1

104 Record Which Stores Joint 2

105 Column of Record 103

106 Memory Address of Record 103

107 Record Which Stores Joint 3

108 Column of Record 104

109 Memory Address of Record 107

110 Column of Record 107

111 Memory Address of Record 104

[Translation done.]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号

特開平10－240741

(43)公開日　平成10年(1998)９月11日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　17/30		Ｇ０６Ｆ　15/419	３１０

審査請求　未請求　請求項の数２　ＯＬ　（全　５　頁）

(21)出願番号　特願平9－46368

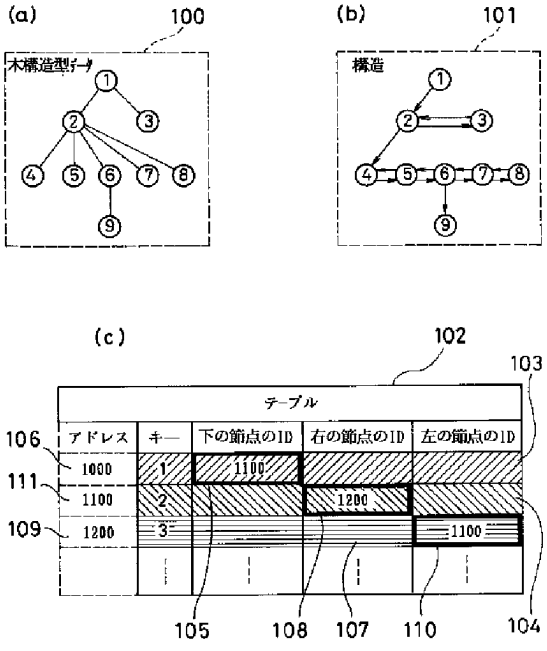
(22)出願日　平成９年(1997)２月28日

(71)出願人　000004226
日本電信電話株式会社
東京都新宿区西新宿三丁目19番２号
(72)発明者　小西　史和
東京都新宿区西新宿三丁目19番２号　日本電信電話株式会社内
(72)発明者　小林　伸幸
東京都新宿区西新宿三丁目19番２号　日本電信電話株式会社内
(72)発明者　夏目　義久
東京都新宿区西新宿三丁目19番２号　日本電信電話株式会社内
(74)代理人　弁理士　三好　秀和　（外１名）

最終頁に続く

(54)【発明の名称】　木構造型データの管理方法

(57)【要約】
【課題】　木構造型データに対する高速な検索が可能であるとともに、木構造型データを効率よく格納し得る木構造型データの管理方法を提供する。
【解決手段】　木構造型データを関係データベースに格納する処理において木構造の節点に当たるデータを関係データベースのレコードとして格納し、この格納されたレコードの格納位置を表すＩＤを設け、該ＩＤをレコードのカラムに格納し、木構造における１つの親とその親に対する複数の子との間の枝を前記複数の子の中の任意の有限個の子との間の枝に限定する。



1

【特許請求の範囲】

【請求項１】　木構造型データを関係データベースに格納する処理において木構造の節点に当たるデータを関係データベースのレコードとして格納し、

この格納されたレコードの格納位置を表すＩＤを設け、該ＩＤをレコードのカラムに格納し、

木構造における１つの親とその親に対する複数の子との間の枝を前記複数の子の中の任意の有限個の子との間の枝に限定することを特徴とする木構造型データの管理方法。

【請求項２】　木構造型データを格納した関係データベースから木構造中の任意の節点から前記節点の子を検索する際に、前記ＩＤで示される格納位置に存在するレコードを検索することを特徴とする請求項１記載の木構造型データの管理方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、木構造型データを格納するとともに、この格納された木構造型データを検索する木構造型データの管理方法に関する。

【０００２】

【従来の技術】木構造型データは、図３に示すような構造を有するデータである。同図において、丸で表したものそれぞれを節点（頂点）と呼ぶ。節点の中の１つを他と区別して根（root）と呼ぶ。節点と節点との間には親子関係が存在し、親と子との節点の間は枝（辺）と呼ばれる線で結ばれている。枝の上側の節点が親であり、下側の節点が子である。根以外の各節点には１つだけ親がある。

【０００３】一方、節点の子の数には制限はなく、１個の親に対してｎ個の子の枝が存在し得る。各節点はいくつの子の節点を持っていてもよいし、子を持たない節点があってもよい。このような規則に従って全ての節点を枝で結んだものが木構造型データである。

【０００４】このような木構造型データを関係データベースで扱う場合、以下に示す方法がある。

【０００５】第１の方法として、図４に示すように、木構造全体を１つのテーブルとして表現する。すなわち、テーブルの１つのレコードが木構造上の１つの節点を表し、テーブルにカラムを木構造の段数分用意し、根からある節点に至る道筋に存在する全ての節点のキーを順番にカラムに格納することで木構造の上下関係を表現する。

【０００６】第２の方法として、図５に示すように、木構造の各節点に、木構造内で一意に識別するためのキーを定め、テーブルの各レコードは各々の節点を表し、テーブルのカラムに各節点のキー、すなわち自分自身のキーと、その直接上位の節点のキーである親の節点のキーをそれぞれ格納することで、木構造における上下関係を表現する。

2

【０００７】

【発明が解決しようとする課題】上述した従来の木構造型データの管理方法では、以下のような問題点があった。

【０００８】（１）第１の方法を用いると、木構造の検索に必要なデータベース操作は少なくて済むが、テーブルに予め全ての段数分カラムを用意する必要があるので、利用しないカラムを多く含み、データの格納効率が非常に悪い。また、木の段数はデータの挿入に伴い増加していく可能性があり、その場合テーブルのカラム数が変化するので、データベースを作り直す必要が出てくる。

【０００９】（２）第２の方法を用いると、木構造をｍ段検索するためには、ｍ回のテーブルに対するデータベース操作（参照操作）が必要である。木構造の段数が深い場合かなりの検索時間を要する。

【００１０】本発明は、上記に鑑みてなされたもので、その目的とするところは、木構造型データに対する高速な検索が可能であるとともに、木構造型データを効率よく格納し得る木構造型データの管理方法を提供することにある。

【００１１】

【課題を解決するための手段】上記目的を達成するため、請求項１記載の本発明は、木構造型データを関係データベースに格納する処理において木構造の節点に当たるデータを関係データベースのレコードとして格納し、この格納されたレコードの格納位置を表すＩＤを設け、該ＩＤをレコードのカラムに格納し、木構造における１つの親とその親に対する複数の子との間の枝を前記複数の子の中の任意の有限個の子との間の枝に限定することを要旨とする。

【００１２】請求項１記載の本発明にあっては、木構造型データにおける各節点に当たるデータを関係データベースのレコードに格納し、この格納したレコードの格納位置を表すＩＤを取得し、レコードのカラムに格納し、木構造における親子関係をＩＤで表現している。また、木構造における親子関係を関係データベースで表現するために、任意の節点を親とした場合の親とその親に対する複数の子との枝を有限個に限定し、有限個数分のＩＤを格納するためのカラムをレコードに用意し、任意の節点を親とした場合の親に対する複数の子同士の間に互いに有限個の枝を定義し、この枝分のＩＤを格納するためのカラムをレコードに設けて、木構造を関係データベースに格納している。

【００１３】また、請求項２記載の本発明は、請求項１記載の発明において、木構造型データを格納した関係データベースから木構造中の任意の節点から前記節点の子を検索する際に、前記ＩＤで示される格納位置に存在するレコードを検索することを要旨とする。

【００１４】請求項２記載の本発明にあっては、関係デ

ータベースに格納された木構造型データに対して任意の節点（レコード）からその子の節点への検索を行う場合に、前記ＩＤが指し示す位置にＩＤを用いて直接該当レコードを検索する。

【００１５】

【発明の実施の形態】以下、図面を用いて本発明の実施の形態について説明する。

【００１６】図１を参照して、本発明の一実施形態に係る木構造型データの管理方法について説明する。

【００１７】まず、図１（ａ）に示す木構造型データ１００を関係データベースに格納する方法を説明する。

【００１８】節点２のように枝が５本ある場合、節点２の子への枝をテーブルに格納するためにはカラムが５個必要となり、節点１のように枝が２本ある場合は、節点１の子への枝をテーブルに格納するためにはカラムが２個必要となり、節点が持つ子への枝の数に応じて必要なカラム数が変化することになる。

【００１９】そこで、本実施形態では、親と子の間の枝を有限個に固定し、木構造型データ１００における親と子の関係を図１（ｂ）に示す構造１０１のように変形させる。このことにより、節点６のように、１つの節点が必要とする枝の数は最高で３本（下矢印、右矢印、左矢印）となり、枝の数を固定することが可能になる。

【００２０】このように有限個のカラムで格納できるようになった構造１０１を図１（ｃ）に示すテーブル１０２に格納する方法を説明する。

【００２１】テーブル１０２には、キーを格納するカラムと、下の節点が格納されているレコードの格納位置を指すＩＤを格納するカラムと、右の節点が格納されているレコードの格納位置を指すＩＤを格納するカラムと、左の節点が格納されているレコードの格納位置を指すＩＤを格納するカラムを用意する。ただし、この例では、テーブルはメモリ上に配置してあるものとし、レコードの格納位置を指すＩＤはメモリアドレスとするが、格納位置を指すＩＤは、格納されている位置を直接に表現できるようなものであれば何でもかまわない。

【００２２】構造１０１のデータをテーブル１０２に格納する処理のステップは以下の通りである。

【００２３】（ステップＳ１）節点１のキーをテーブル１０２のレコード１０３に格納する。

【００２４】（ステップＳ２）節点２のキーをテーブル１０２のレコード１０４に格納する。

【００２５】（ステップＳ３）レコード１０３の下の節点のＩＤ格納カラム１０５にレコード１０４のメモリアドレス１１１を格納する。

【００２６】（ステップＳ４）節点３のキーをテーブル１０２のレコード１０７に格納する。

【００２７】（ステップＳ５）レコード１０４の右の節点のＩＤ格納カラム１０８にレコード１０７のメモリアドレス１０９を格納する。

【００２８】（ステップＳ６）レコード１０７の左の節点のＩＤ格納カラム１１０にレコード１０４のメモリアドレス１１１を格納する。

【００２９】上記のステップを構造１０１の全ての節点について行う。ただし、節点８のように、右もしくは左もしくは下の節点が存在しない場合は、それ以上節点が存在しないことが識別できる値を格納する。

【００３０】次に、テーブル１０２に対する検索方法を説明する。

【００３１】テーブル１０２に対して、節点１（キーは１とする）の子である節点３（キーは３とする）を検索する処理のステップは以下の通りである。

【００３２】（ステップＳ１）節点１が格納されているレコード１０３の、下の節点のＩＤ格納カラムに格納されているメモリアドレス１０５を用いて、レコード１０４にジャンプする。

【００３３】（ステップＳ２）レコード１０４のキー２と与えられた節点３のキー３を比較する。

【００３４】（ステップＳ３）キーが一致しないので、レコード１０４の、右の節点のＩＤ格納カラム１０８に格納されているメモリアドレス１０９を用いて、レコード１０７にジャンプする。

【００３５】（ステップＳ４）レコード１０７のキー３と与えられた節点３のキー３を比較する。

【００３６】（ステップＳ５）キーが一致したので、検索処理を終了する。

【００３７】上記のステップを目的の節点が見つかるまで続ける。もし、上記のそれ以上節点が存在していないことが識別できる値が右の節点のＩＤ格納カラムに格納されていた場合、そのキーと一致する節点がデータベース内に存在しないことを意味する。

【００３８】次に、図２を参照して、本発明の他の実施形態について説明する。

【００３９】図２（ａ）に示す木構造型データ２００を関係データベースに格納する方法を説明する。

【００４０】本実施形態においては、親と子との間の枝を有限個に固定するために、木構造型データ２００における親と子の関係を図２（ｂ）に示す構造２０１のように変形させる。このことにより、節点３のように、１つの節点が必要とする枝の数は最高で５本（左下矢印、下真ん中矢印、右下矢印、右矢印、左矢印）となり、枝の数を固定することが可能となる。

【００４１】次に、上記により有限個のカラムで格納できるようなった構造２０１をテーブルに格納する方法は図１の実施形態と同様である。ただし、構造２０１における節点５から節点９までのキーを予め昇順もしくは降順に並べ替えを行っておく。

【００４２】次に、テーブルに対する検索方法も図１の実施形態と同様である。ただし、任意の節点からその子を検索する際に、下真ん中矢印の指す節点のキー値とで

比較を行ってから、３つの下向きＩＤのレコードにジャンプするかの決定を行う。

【００４３】上述したように、本発明の木構造型データの管理方法では、木構造における各節点をレコードに格納し、各節点間の枝（関係）をレコードの格納位置を指すＩＤとしてレコードのカラムに格納することで、木に対する検索を行う場合に、索引経由の検索を行うことなく、ＩＤの指すレコードに直接ジャンプすることで、高速な検索を可能にする。

【００４４】また、木構造における１：ｎの親子関係を、親から子への枝を有限個に限定し、子同士を互いに枝で結ぶことで、有限（固定）個のカラムで木構造を表現するので、利用しないカラムが無くなり効率的な格納が行える。

【００４５】

【発明の効果】以上説明したように、本発明によれば、木構造における親子関係を各節が格納されている格納位置を表すＩＤを用いて表現しているため、任意の節点の子を検索する場合に、前記ＩＤを用いることで物理的レコードの位置を取得することが可能となり、これにより直接前記レコードを求めることが可能となる。よって、木構造型データの検索に、データベース操作（参照操作）を用いる必要がない。

【００４６】また、任意の節点を親とした場合の、前記親と前記親に対する複数の子との間の枝を有限個に限定でき、前記親に対する複数の子同士の間の枝も有限個に限定できるので、テーブルに必要なカラムは、木構造の段数に関係なく固定にでき、利用しないカラムが無くなり格納効率が非常によくなる。

【００４７】また、テーブルに必要なカラムは固定数（上記有限個）であるので、木構造の段数が増えた場合などにおいて、新たにデータベースを作り直す必要がなくなる。

【図面の簡単な説明】

【図１】本発明の一実施形態に係る木構造型データの管理方法を説明するための図である。

【図２】本発明の他の実施形態を説明するための図である。

【図３】木構造型データを説明するための図である。

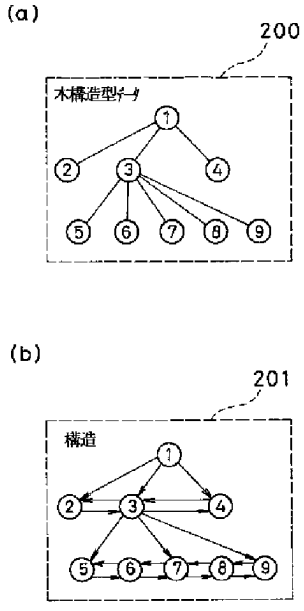
【図４】従来の木構造型データの管理方法を説明するための図である。

【図５】従来の木構造型データの管理方法を説明するための図である。

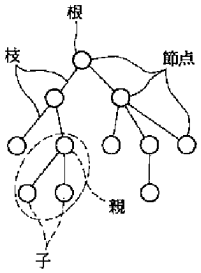
【符号の説明】

１００，２００　木構造型データ
１０１，２０１　変換後の木構造型データ
１０２　テーブル
１０３　節点１を格納するレコード
１０４　節点２を格納するレコード
１０５　レコード１０３のカラム
１０６　レコード１０３のメモリアドレス
１０７　節点３を格納するレコード
１０８　レコード１０４のカラム
１０９　レコード１０７のメモリアドレス
１１０　レコード１０７のカラム
１１１　レコード１０４のメモリアドレス

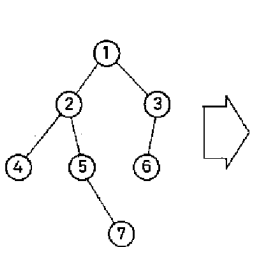
【図２】



【図３】



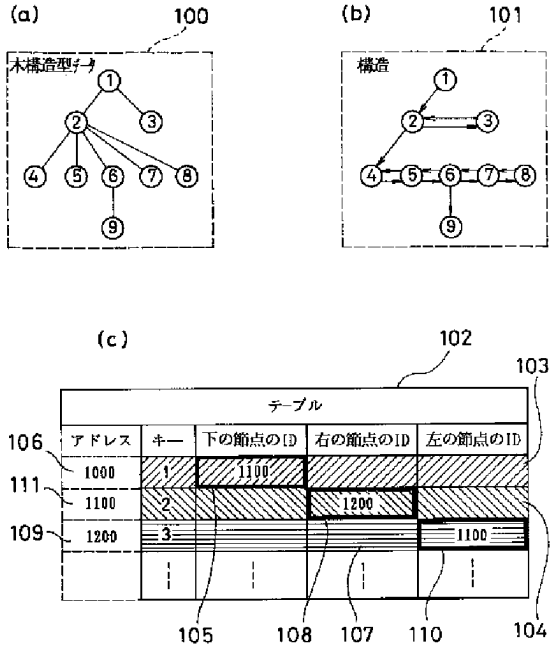
【図４】



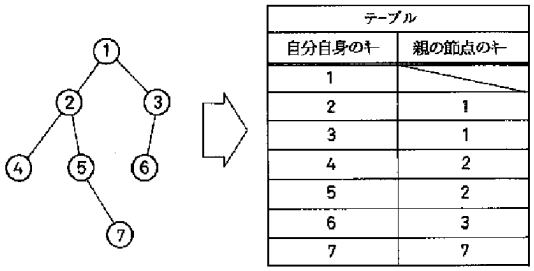
テーブル

1段目	2段目	3段目	4段目
1			
1	2		
1	3		
1	2	4	
1	2	5	
1	3	6	
1	2	5	7

【図１】



【図５】



テーブル	
自分自身のキー	親の節点のキー
1	
2	1
3	1
4	2
5	2
6	3
7	7

フロントページの続き

(72)発明者　西岡　秀一
東京都新宿区西新宿三丁目19番２号　日本
電信電話株式会社内